

# Correlation Clustering with Low-Rank Matrices

Nate Veldt<sup>1</sup>, Anthony Wirth<sup>2</sup> and David Gleich<sup>3</sup>

<sup>1</sup>Department of Mathematics, Purdue University

<sup>2</sup>Department of Computer Science, Purdue University

<sup>3</sup>Department of Computing and Information Systems, The University of Melbourne

November 23, 2016

## Abstract

Correlation clustering is a technique for aggregating data based on qualitative information about which pairs of objects are labeled ‘similar’ or ‘dissimilar.’ Because the optimization problem is NP-hard, much of the previous literature focuses on finding approximation algorithms. In this paper we explore a new approach to correlation clustering by considering how to solve the problem when the data to be clustered can be represented by a low-rank matrix. Many real-world datasets are known to be inherently low-dimensional, and our goal is to establish a tractable approach to correlation clustering in this important setting. We prove in particular that correlation clustering can be solved in polynomial time when the underlying matrix is positive semidefinite with small constant rank, but that the task remains NP-hard in the presence of even one negative eigenvalue. Based on our theoretical results, we develop an algorithm for efficiently solving low-rank positive semidefinite correlation clustering by employing a procedure for zonotope vertex enumeration. We demonstrate the effectiveness and speed of our algorithm by using it to solve several clustering problems on both synthetic and real-world data.

## 1 Introduction

Correlation clustering is a method for partitioning a dataset based on pairwise information that indicates whether pairs of objects in the given dataset are ‘similar’ or ‘dissimilar.’ The problem was first introduced by Bansal, Blum, and Chawla [3] and has since been widely studied both for its interesting theoretical properties as well as its success in real-world clustering applications. Typically correlation clustering is cast as a graph optimization problem where the nodes of a graph represent objects from the dataset. In its most basic form, the graph is assumed to be complete and unweighted, with each edge being labeled ‘+’ or

‘−’ depending on whether the two nodes are ‘similar’ or ‘dissimilar.’ Given this input, the objective is to partition the graph in a way that maximizes the number of agreements, where an agreement is a ‘+’ edge that is included inside a cluster, or a ‘−’ edge that links nodes in different clusters. An equivalent objective, though more difficult to approximate, is the goal of minimizing disagreements, i.e., ‘similar’ nodes that are separated or ‘dissimilar’ nodes that are clustered together. A more general form of correlation clustering associates each pair of objects with not only a label but also a weight indicating how similar or dissimilar the two objects are. In this case, the goal is to maximize the weight of agreements or minimize the weight of disagreements.

One attractive property of this clustering approach is that the number of clusters formed is determined automatically by optimizing the objective function, rather than being a required user input. This makes correlation clustering particularly useful for applications where the optimal number of clusters to form is not known ahead of time. In practice, correlation clustering has been applied in a wide variety of disciplines to solve problems such as cross-lingual link detection [21], gene clustering [4], image segmentation [10], and record linkage in natural language processing [13].

Because correlation clustering is NP-hard [3], much of the previous literature has focused on developing approximation algorithms. In this paper, we consider a new approach, exploring instances of the problem where the weighted labels can be represented by a low-rank matrix. Our motivation for addressing this special case is twofold.

First, by exploring special families of instances, we aim to develop a new way of dealing with the intractability of correlation clustering. This furthers the understanding of the theory of correlation clustering, developing more refined notions of the problem’s complexity.

Second, we wish to apply the framework of correlation clustering in the analysis of low-dimensional datasets. It is well known that real-world datasets are often inherently low-dimensional or can at least be well-approximated by an embedding into a low-dimensional manifold despite residing in a higher-dimensional space. Many other successful techniques have been applied to solve clustering problems under this assumption. A common approach in this context is to take the low-dimensional embedding of the data and partition the data using Lloyd’s  $k$ -MEANS algorithm, which clusters items based on the Euclidean distance between points. We are interested instead in applying the framework of correlation clustering to data analysis problems where it is possible to obtain good low-dimensional representations of datasets.

**Our Contributions:** In this paper we prove that correlation clustering can be solved in polynomial time when the similarity labels can be represented by a positive semidefinite matrix of low rank. We also show that the problem remains NP-hard when the underlying matrix has even one negative eigenvalue. To solve correlation clustering problems in practice, we implement an algorithm called ZONOC based on the randomized zonotope vertex enumeration procedure of Stinson, Gleich, and Constantine [19]. This algorithm is capable of optimally solving low-rank positive semidefinite correlation clustering. It is most useful,

however, when it is truncated at a fixed number of iterations in order to quickly obtain a very good approximation – though sans formal guarantee – to the optimal solution. We demonstrate the effectiveness of ZONOC by obtaining clusterings for both synthetic and real-world datasets, including social network datasets and search query data for well-known computer science conferences.

## 2 Problem Statement

We begin with the standard approach to correlation clustering by considering a graph with  $n$  nodes where edges are labeled either ‘+’ or ‘−’. Typically the correlation clustering objective is cast as an integer linear program in the following way. For every pair of nodes  $i$  and  $j$  we are given two nonnegative weights,  $w_{ij}^+$  and  $w_{ij}^-$ , which indicate a score for how similar the two nodes are and a score for how dissimilar they are respectively. Traditionally, we assume that only one of these weights is nonzero (if not, they can be adjusted so this is the case without changing the objective function). For every pair of nodes  $i, j$  we introduce a binary variable  $d_{ij}$  such that

$$d_{ij} = \begin{cases} 0 & \text{if } i \text{ and } j \text{ are clustered together} \\ 1 & \text{if } i \text{ and } j \text{ are separated} \end{cases}$$

In other words,  $d_{ij} = 1$  indicates we have cut the edge between nodes  $i$  and  $j$ . The maximization version of correlation clustering is given by the following ILP (integer linear program). We include triangle constraints on the  $d_{ij}$  variables to guarantee that they define a valid clustering on the nodes.

$$\begin{aligned} & \text{maximize} && \sum_{i < j} w_{ij}^+ (1 - d_{ij}) + \sum_{i < j} w_{ij}^- d_{ij}, \\ & \text{subject to} && d_{ij} \in \{0, 1\}, \\ & && d_{ik} \leq d_{ij} + d_{jk} \text{ for all } i, j, k. \end{aligned} \tag{1}$$

The first term counts the weight of agreements from clustering similar nodes together, and the second counts the weight of disagreements from dissimilar nodes that are clustered apart.

**Correlation Clustering Matrix** For convenience, we encode the weights of a correlation clustering problem into a matrix  $\mathbf{A}$  by defining  $A_{ij} = w_{ij}^+ - w_{ij}^-$ . We think of  $\mathbf{A}$  as the adjacency matrix of a graph that has both positive and negative edges where the sign of the edge weight indicates similarity or dissimilarity. The correlation clustering problem thus becomes finding a partition of this signed graph where negative edges are cut as much as possible and positive edges are in the interior of the clusters as much as possible. We can express the objective function in terms of  $\mathbf{A}$  as

$$\text{maximize} \quad - \sum_{i < j} A_{ij} d_{ij} + \sum_{i < j} w_{ij}^+.$$

The second term is only a constant, so if we want to solve this problem optimally we can focus on just solving the ILP:

$$\begin{aligned} & \text{maximize} && -\sum_{i<j} A_{ij}d_{ij} \\ & \text{subject to} && d_{ij} \in \{0,1\}, \\ & && d_{ik} \leq d_{ij} + d_{jk} \text{ for all } i,j,k. \end{aligned} \tag{2}$$

We can provide an alternative formulation of the correlation clustering objective by introducing an indicator vector  $\mathbf{x}_i \in \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \dots, \mathbf{e}_n\}$  for each node  $i$ , where  $\mathbf{e}_j$  is the  $j$ th standard basis vector in  $\mathbb{R}^n$ . This indicates which cluster node  $i$  belongs to. Unless each node ends up in its own singleton cluster, some of these basis vectors will be unused. We can then make the substitution  $d_{ij} = 1 - \mathbf{x}_i^T \mathbf{x}_j$ , since  $\mathbf{x}_i^T \mathbf{x}_j$  will be one if both nodes are in the same cluster but will be zero otherwise. After making this substitution and dropping a constant term in the objective, the problem becomes

$$\begin{aligned} & \text{maximize} && \sum_{i<j} A_{ij} \mathbf{x}_i^T \mathbf{x}_j \\ & \text{subject to} && \mathbf{x}_i \in \{\mathbf{e}_1, \dots, \mathbf{e}_n\} \text{ for all } i = 1, \dots, n. \end{aligned} \tag{3}$$

We remark that the output clustering doesn't depend on the diagonal of the matrix  $\mathbf{A}$ , as we will always cluster a node with itself. For this reason, in any algorithm implementations and applications we can select any diagonal for the matrix without changing the optimal clustering.

### 3 Main Theoretical Results

In this section, we present new results on the complexity of correlation clustering under low-rank assumptions. In particular, correlation clustering remains NP-hard when the underlying matrix has even one negative eigenvalue. However, when the underlying matrix is positive semidefinite of rank  $d$ , the problem can be solved in polynomial time. We will use the second result to develop an efficient algorithm for positive semidefinite correlation clustering in Section 4 and show several applications and experimental results in Section 5.

#### 3.1 Positive Semidefinite CC

The simplest case to consider is when  $\mathbf{A}$  is rank-1 with one positive eigenvalue. Because  $\mathbf{A}$  is symmetric, we can express it as  $\mathbf{A} = \mathbf{v}\mathbf{v}^T$  for some  $\mathbf{v} \in \mathbb{R}^n$ . In this case a perfect clustering always exists and is easy to find. One cluster includes all nodes with negative entries in  $\mathbf{v}$ , while the other includes those with positive entries. Nodes  $i$  and  $j$  are similar if and only if  $A_{ij} > 0$ , which is true if and only if entries  $i$  and  $j$  of the vector  $\mathbf{v}$  have the same sign. So this simple two clustering perfectly agrees with the similarity labels and forms an ideal clustering. This version of the problem can be shown to be equivalent to maximizing a quadratic form in binary variables:

$$\begin{aligned} & \text{maximize} && \mathbf{x}^T \mathbf{A} \mathbf{x} \\ & \text{subject to} && \mathbf{x} \in \{-1, 1\}^n \end{aligned}$$

under the special assumption that  $\mathbf{A}$  is rank 1. It is known that this problem can be solved in polynomial time for any fixed low rank  $d$  [12]. While this gives us a nice result for correlation clustering on rank-1 matrices, it does not generalize to higher ranks as it only can partition a graph into exactly two clusters.

If the matrix  $\mathbf{A}$  is positive semidefinite but of rank  $d > 1$ , there is no guarantee that a perfect partitioning exists, and the optimal clustering may have more than two clusters. We still begin by expressing  $\mathbf{A}$  in terms of low rank factors, i.e.,  $\mathbf{A} = \mathbf{V}\mathbf{V}^T$  for some  $\mathbf{V} \in \mathbb{R}^{n \times d}$ . Each node in the signed graph can now be associated with one of the row vectors  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^{d \times 1}$  of  $\mathbf{V}$ . The similarity scores between nodes  $i$  and  $j$  are given by  $A_{ij} = \mathbf{v}_i^T \mathbf{v}_j$ , so we can view this version of correlation clustering as a vector partitioning problem where we must separate  $n$  points or vectors in  $\mathbb{R}^d$  based on similarity scores given by inner products of the vectors. We formalize this with our first theorem:

**THEOREM 1** *If  $\mathbf{A} = \mathbf{V}\mathbf{V}^T$  for  $\mathbf{V} \in \mathbb{R}^{n \times d}$ , then problem (3) can be solved by partitioning the row vectors  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^{d \times 1}$  of  $\mathbf{V}$  into  $d + 1$  clusters  $\{C_1, C_2, \dots, C_{d+1}\}$  to solve*

$$\text{maximize} \quad \sum_{i=1}^{d+1} \|S_i\|_2^2, \quad (4)$$

where we refer to the vector  $S_i = \sum_{\mathbf{v} \in C_i} \mathbf{v}$  as the **sum point** of the  $i^{\text{th}}$  cluster (in an empty cluster, defined to be the zero vector).

We will show in two parts that when  $\mathbf{A} = \mathbf{V}\mathbf{V}^T$ , the clustering that maximizes (3) also maximizes (4). First we prove that (3) is equivalent to maximizing the sum of squared norms of sum points, where the maximization is taken over any possible clustering. We will then show that the objective function is maximized by a clustering with  $d + 1$  or fewer clusters.

**Step 1: Maximizing a function on sum points** By doubling the objective function in (3) and adding the constant  $\sum_{i=1}^n \mathbf{v}_i^T \mathbf{v}_i$ , we obtain a related objective function that is maximized by the same clustering:

$$2 \sum_{i < j} \mathbf{v}_i^T \mathbf{v}_j (\mathbf{x}_i^T \mathbf{x}_j) + \sum_{i=1}^n \mathbf{v}_i^T \mathbf{v}_i = \sum_{i=1}^n \sum_{j=1}^n \mathbf{v}_i^T \mathbf{v}_j (\mathbf{x}_i^T \mathbf{x}_j). \quad (5)$$

Recall that  $\mathbf{x}_i$  and  $\mathbf{x}_j$  are indicator vectors that identify which clusters nodes  $i$  and  $j$  belong to. Note that the right-hand side of equation (5) only counts the product  $\mathbf{v}_i^T \mathbf{v}_j$  when  $\mathbf{x}_i^T \mathbf{x}_j = 1$ , (i.e., for pairs  $i, j$  that are in the same cluster), so we are restricting our attention to inner products between vectors that belong to the same cluster. The contribution to the objective from cluster  $C_k$  is

$$\sum_{a \in C_k} \sum_{b \in C_k} \mathbf{v}_a^T \mathbf{v}_b = S_k^T S_k = \|S_k\|_2^2.$$

Summing over all clusters completes the first step of the proof.

**Step 2: Bounding the number of clusters** To see that the number of clusters is bounded, observe that in the optimal clustering all the sum points must have pairwise non-positive dot products. Otherwise there exist distinct clusters  $C_i$  and  $C_j$  such that  $S_i^T S_j > 0$ , and therefore

$$(S_i + S_j)^T (S_i + S_j) = S_i^T S_i + 2S_i^T S_j + S_j^T S_j > S_i^T S_i + S_j^T S_j.$$

This implies that we can get a better clustering by combining  $C_i$  and  $C_j$ . Note that if we further assume that we are looking for the clustering which optimizes the objective by forming as few clusters as possible, we can assume that the inner products between distinct sum points is always negative.

Note that if two sum points have inner product zero, we can merge their respective clusters without changing the objective function score. While there may be multiple clusterings which maximize (3), we only need to find one, so it is sufficient find the optimal clustering with the minimum number of clusters. In this case we can ignore clusterings with sum points that have inner product zero. This effectively restricts our search to clusterings where all sum points have pairwise negative dot products. The upper bound of  $d + 1$  clusters then follows from the fact that the maximum number of vectors in  $\mathbb{R}^d$  with pairwise negative inner products is  $d + 1$  (Lemma 8 of Rankin [18]).  $\square$

It is worth noting that despite a significant difference in motivation, our new objective function (4) is nearly identical to one used by Newman as a means to approximately solve maximum modularity clustering [14]. This gives an interesting new connection between two clustering techniques that were not previously known to be related. We direct the reader to Newman and Zhang’s work for more information on modularity [22, 14].

The importance of Theorem 1 is that it expresses the low-rank positive semidefinite correlation clustering problem as a convex functional on sums of vectors in  $\mathbb{R}^d$ . Our problem is therefore an instance of the well studied vector partitioning problem [15, 9]. Onn and Schulman showed that for dimension  $d$  and a fixed number of clusters  $p$ , this problem can be solved using  $O(n^{d(p-1)-1})$  arithmetic operations and queries to an oracle function by exploring the vertices of a  $d^2$ -dimensional polytope called the *signing zonotope* [15]. Combining this result with our Theorem 1 above gives the following corollary:

**COROLLARY 2** *Correlation clustering with rank- $d$  positive semidefinite matrices (PSD-CC) is a special case of the vector partition problem with  $d + 1$  clusters, and is therefore solvable in polynomial time.*

We will later show how to construct a polynomial-time algorithm for PSD-CC by reviewing the results of Onn and Schulman [15]. Before concluding this subsection, we present our second theorem, that highlights an important geometric feature satisfied by the optimal clustering. We include it to provide an intuitive first look at how to solve the problem in polynomial time.

**THEOREM 3** *In the clustering  $\mathcal{C}_{opt}$ , which maximizes (4), the  $n$  row vectors of  $V$  will be separated into distinct convex cones that intersect only at the origin.*

More precisely, if vectors  $\mathbf{v}_{x_1}, \mathbf{v}_{x_2}, \dots, \mathbf{v}_{x_k}$  are all in the same cluster  $C_x$  in  $\mathbf{C}_{opt}$ , and  $\mathbf{v}_y \in \mathbb{R}^d$  is another row vector that satisfies  $\mathbf{v}_y = \sum_{i=1}^k c_i \mathbf{v}_{x_i}$  for  $c_i \in \mathbb{R}_0^+$ , then  $\mathbf{v}_y$  is also in cluster  $C_x$ .

**Proof** First notice that in  $\mathbf{C}_{opt}$ , every vector  $\mathbf{v}$  must be more similar to its own sum point than to any sum point of a different cluster. To see this, assume that  $\mathbf{v}$  is in cluster  $C_i$  with sum point  $S_i$ , but  $\mathbf{v}$  is more similar to another sum point  $S_j$ , i.e.

$$\mathbf{v}^T S_i < \mathbf{v}^T S_j.$$

The contribution to the objective from the two sum points is  $S_i^T S_i + S_j^T S_j$ . If we move  $\mathbf{v}$  from cluster  $C_i$  to cluster  $C_j$ , the contribution to the objective for the two new clusters is

$$\begin{aligned} (S_i - \mathbf{v})^T (S_i - \mathbf{v}) + (S_j + \mathbf{v})^T (S_j + \mathbf{v}) = \\ S_i^T S_i - 2\mathbf{v}^T S_i + \mathbf{v}^T \mathbf{v} + S_j^T S_j + 2\mathbf{v}^T S_j + \mathbf{v}^T \mathbf{v} \end{aligned}$$

which is a higher score since  $\mathbf{v}^T S_i < \mathbf{v}^T S_j$ , contradicting the optimality of the first clustering. So in the optimal clustering every point is more similar to its own sum point than any other sum point.

Given this first observation we will now prove the main result of the theorem by contradiction. Assume that we have  $k$  points  $\mathbf{v}_{x_1}, \mathbf{v}_{x_2}, \dots, \mathbf{v}_{x_k}$  that in  $\mathbf{C}_{opt}$  are in cluster  $C_x$  with sum point  $S_x$ . Let  $\mathbf{v}_y$  be another point in the dataset such that  $\mathbf{v}_y = \sum_{i=1}^k c_i \mathbf{v}_{x_i}$  where  $c_i > 0$  for  $i = 1, 2, \dots, k$ , and assume that  $\mathbf{v}_y$  is in a different cluster  $C_y$  that has sum point  $S_y$ . By our first observation, every point in  $C_x$  must be more similar to  $S_x$  than  $S_y$ , so for  $1 \leq i \leq k$  we have that  $\mathbf{v}_{x_i}^T S_x > \mathbf{v}_{x_i}^T S_y$ , which implies  $c_i \mathbf{v}_{x_i}^T S_x > c_i \mathbf{v}_{x_i}^T S_y$ , for any positive scalar  $c_i$ . It follows that

$$\mathbf{v}_y^T S_x = \sum_{i=1}^k c_i \mathbf{v}_{x_i}^T S_x > \sum_{i=1}^k c_i \mathbf{v}_{x_i}^T S_y = \mathbf{v}_y^T S_y$$

which indicates that  $\mathbf{v}_y$  is more similar to  $S_x$  than to the sum point of the cluster to which it belongs, which is a contradiction.  $\square$

Theorem 3 implies that we can find an optimal clustering in polynomial time by checking every possible partitioning of the vectors into convex cones. By Theorem 2.7 of Klee [11], any two cones can be separated by a hyperplane through the origin. Furthermore, Cover [6] proved that for any  $n$  points in  $\mathbb{R}^d$  there are  $O(n^{d-1})$  such hyperplanes that split the points into two groups. This means that to cluster the points into  $d+1$  convex cones, we must choose  $\binom{d+1}{2}$  of the  $O(n^{d-1})$  hyperplanes so that each distinct pair of clusters is separated by a hyperplane. This gives us a total of  $O(n^{(d-1)\binom{d+1}{2}})$  ways to cluster the points into  $d+1$  convex cones, which we can enumerate in polynomial time. When  $d=2$ , each of the  $O(n)$  hyperplanes corresponds to a line that connects the origin and one of the  $n$  points. Since there are at most three clusters in this case, we can solve the problem by testing  $O(n^3)$  triplets of points and evaluating

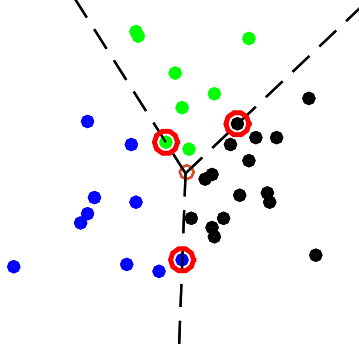


Figure 1: The three-clustering that solves a small correlation clustering problem. Each cluster is shown by a different color, and can be delimited on each side by lines through the origin. By selecting the right combination of three points (circled in red), we will be able to find the optimal clustering.

the objective for each corresponding clustering. A visualization of this process is shown in Figure 1. This separating-hyperplane procedure is related to the method proposed by Onn and Schulman [15] which we review in Section 4, but for any dimension  $d$  greater than 2 it is significantly less efficient.

### 3.2 Rank-1 Negative Semidefinite

When  $\mathbf{A}$  has rank-1 and its nonzero eigenvalue is negative, we know  $\mathbf{A} = -\mathbf{v}\mathbf{v}^T$  for some  $\mathbf{v} \in \mathbb{R}^d$ . This changes the objective (4) in three ways: the negative sign converts the maximization to a minimization, the entries of  $\mathbf{v}$  are real numbers rather than row vectors, and the upper bound of  $d + 1$  clusters no longer applies. Solving the Rank One Negative Eigenvalue Correlation Clustering problem (RONE-CC) is therefore equivalent to

$$\min_{C,k} \sum_{i=1}^k \left( \sum_{v \in C_i} v \right)^2 \quad (6)$$

Where  $C = \{C_1, C_2, \dots, C_k\}$  is a clustering of the nodes.

**THEOREM 4** *Rank-one Negative Eigenvalue Correlation Clustering is NP-Complete.*

**Proof** Given that general correlation clustering is NP-Complete, we know that the decision version of RONE CC must also be in NP. To show the problem is NP-hard, we can use a reduction from the partition problem, one of the NP-hard problems listed by Garey and Johnson [7]. For this problem we are given a multiset of  $n$  positive integers and seek to partition the set into subsets of equal sum. Consider a multiset of  $n$  positive integers and multiply each integer by two so that the smallest integer  $s$  and the sum of the integers  $B$  are both even. Letting  $M = B/2 - s/2$ , we add two copies of  $-M$  to the input: now the total



sum of the input integers is  $s$ . For the proof we assume that the positive integers can be split into two subsets of sum  $B/2$ . Assume the positive integers can be split into two subsets of sum  $B/2$ . If we include exactly one copy of the  $-M$  values in each of the two partitions, then each sums to  $B/2 - M = s/2$ . We leave it as an exercise to show that any other partitioning of the integers will result in a clustering with higher RONE-CC objective score. Therefore, RONE-CC will perfectly partition the  $n$  positive integers into subsets of equal sum if and only if such a perfect partition exists.

We can show that the optimal solution to RONE-CC will perfectly partition the  $n$  positive integers into two subsets of equal sum, if and only if such a perfect partition exists. Assume the positive integers can be split into two subsets of sum  $B/2$ . If we include exactly one copy of the  $-M$  values in each of the two partitions, then each sums to  $B/2 - M = s/2$ . The RONE-CC objective corresponding to this clustering is  $2(s/2)^2 = s^2/2$ . We now show that every other clustering yields a worse objective value. Clearly, a clustering with one copy of  $-M$  in each cluster but no equal split of the positives, has objective function  $> s^2/2$ .

If we cluster all integers together, the sum is  $s$ , and the objective would be  $s^2 > s^2/2$ . If on the other hand we consider a two-clustering where both  $-M$  values are in the same partitioning, or a clustering with more than two clusters, then there must exist some cluster with only positive integers. This cluster has sum at least  $s$ , leading to an objective of at least  $s^2$ . The best option is therefore to form two clusters, each of which contains one of the  $-M$  values and a subset of the  $n$  positive integers summing to  $B/2$ .  $\square$

## 4 Algorithms

In this section we show how to obtain a polynomial-time algorithm for solving PSD-CC. We first review the results of Onn and Schulman [15], which establish the existence of a polynomial-time algorithm, by analyzing the properties of a  $d^2$ -dimensional polytope called the signing zonotope. We then combine this with a vertex-enumeration procedure developed by Stinson, Constantine, and Gleich [19].

### 4.1 Signing Zonotope

A zonotope is the linear projection of a high-dimensional hypercube into a lower-dimensional vector space. We are primarily concerned with the *signing zonotope* introduced by Onn and Schulman [15], whose vertices directly correspond to clusterings of the  $n$  vectors of a vector partition problem.

Consider a set  $S_V$  of  $n$  vectors  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^{d \times 1}$  in an instance of the vector partition problem. A *signing* of these vectors is defined to be a vector  $\sigma = (\sigma_{a,b}^i) \in \{-1, 1\}^M$ , where  $M = n \binom{d+1}{2}$ . Each entry  $\sigma_{a,b}^i$  uniquely corresponds to a triplet  $(\mathbf{v}_i, a, b)$ , where  $\mathbf{v}_i$  is one of the data points we are clustering and  $1 \leq a < b \leq (d+1)$  are the indices for two distinct clusters in a

$(d+1)$ -clustering of the  $n$  vectors. If  $b < a$ , we define  $\sigma_{a,b}^i = -\sigma_{b,a}^i$  and associate each signing with a matrix  $\mathbf{T}_\sigma$ :

$$\mathbf{T}_\sigma = \sum_{i=1}^n \sum_{1 \leq a < b \leq d+1} \sigma_{a,b}^i \mathbf{v}_i \cdot (\mathbf{e}_a - \mathbf{e}_b)^T \in \mathbb{R}^{d \times (d+1)}, \quad (7)$$

where  $\mathbf{e}_a, \mathbf{e}_b \in \mathbb{R}^{(d+1) \times 1}$  are the  $a^{\text{th}}$  and  $b^{\text{th}}$  standard basis vector respectively. By construction, the row sum of  $\mathbf{T}_\sigma$  will be the zero vector, so if we are given the first  $d$  columns of the matrix we will be able to recover the last column even if it is not given explicitly. We associate with each signing  $\sigma$  a vector  $Z_\sigma$  of length  $d^2$  made by stacking the first  $d$  columns of  $\mathbf{T}_\sigma$  on top of one another. Here we will refer to this vector as the Z-vector of  $\sigma$ . A signing  $\sigma$  is said to be *extremal* if its Z-vector is a vertex of the signing zonotope, which we define below. Furthermore, Onn and Schulman proved that for every vertex  $v$  of the zonotope, there exists exactly one extremal signing  $\sigma$  such that  $v$  is the Z-vector of  $\sigma$ .

The *signing zonotope*  $\mathcal{Z}$  for this instance of the vector partition problem is defined to be

$$\mathcal{Z} = \text{conv}\{Z_\sigma : \sigma \text{ is a signing of } S_V\}.$$

In other words,  $\mathcal{Z}$  is the convex hull of all  $2^M$  Z-vectors of signings of  $S_V$ . We now state two important results established by Onn and Schulman [15] about the signing zonotope:

**THEOREM 5** *The following properties hold regarding the signing zonotope  $\mathcal{Z}$  introduced above:*

1. *Each vertex of  $\mathcal{Z}$  can be mapped to a clustering of the  $n$  vectors in  $S_V$ , where each cluster is contained in one of  $d+1$  convex cones. Additionally, there exists an extremal signing that maps to the clustering which optimizes the objective of the vector partition problem.*
2.  *$\mathcal{Z}$  has at most  $O(n^{d^2-1})$  vertices.*

All we need then to solve the vector partition problem, and hence PSD-CC, is to iterate through each extremal signing of  $\mathcal{Z}$ , obtain the clustering it corresponds to, and calculate the objective (4) for that clustering. At the end we output the clustering with the maximum objective value.

The procedure for associating an extremal signing with a clustering of  $S_V$  is given in Proposition 2.3 of Onn and Schulman's work [15]. This states that for all  $i = 1, 2, \dots, n$ , there exists a unique index  $1 \leq c_i \leq d+1$  such that  $\sigma_{c_i,k}^i = 1$  for all  $k \neq c_i$ . Thus vector  $\mathbf{v}_i$  belongs to cluster number  $c_i$  in the optimal clustering. We can interpret this fact in light of Theorem 3. Recall that according to Klee [11], between each pair of clusters  $C_a$  and  $C_b$  there exists a hyperplane that separates  $\mathbb{R}^d$  into two half spaces such that  $C_a$  is in one half and  $C_b$  is in the other. Each extremal signing  $\sigma$  encodes information about which side of the separating hyperplane each vector  $\mathbf{v}_i$  is on. For example,  $\sigma_{a,b}^i = 1$

indicates that vector  $\mathbf{v}_i$  is the same half space as all vectors in cluster  $C_a$ . More importantly, this tells us that  $\mathbf{v}_i$  is *not* in the same half space as cluster  $C_b$ , so we rule out the possibility that  $\mathbf{v}_i$  is in cluster  $C_b$ . On the other hand,  $\sigma_{a,b}^i = -1$  indicates that  $\mathbf{v}_i$  is on the same side as cluster  $C_b$ , eliminating the possibility that  $\mathbf{v}_i$  is in  $C_a$ . If we consider all entries of  $\sigma$ , Onn and Schulman’s proposition effectively tells us that there is only one cluster that will not be ruled out by this process, so by default this is the cluster where  $\mathbf{v}_i$  is located. From this elimination process we recover a  $(d + 1)$ -clustering of the vectors. With this we are now able to state the exact runtime for solving PSD-CC.

**THEOREM 6** *The fixed-rank positive semidefinite correlation clustering problem can be solved in  $O(n^{d^2})$  time.*

**Proof** Relying on previous complexity and algorithmic results regarding zonotopes, in Corollary 3.3, Onn and Schulman establish that the  $(d + 1)$ -vector partition problem can be solved with  $O(n^{d^2-1})$  operations and queries to an oracle for evaluating the convex objective functional [15]. We now show that it takes  $O(n)$  operations to evaluate the oracle function for each of the  $O(n^{d^2-1})$  extremal signings. In our case, the complexity of the oracle is the time it takes to evaluate summation (4) for a given extremal signing  $\sigma$ . Treating  $d$  as a fixed constant, this procedure involves inspecting the  $M = O(n)$  entries of  $\sigma$  to identify a clustering, and  $O(n)$  operations to add vectors in each cluster to obtain the sum points. We require only a constant number of operations to take dot products of the sum points and add the results, so the evaluation process takes  $O(n)$  time, and hence the overall process  $O(n^{d^2})$  time.

## 4.2 Practical Algorithm

Though theoretically polynomial time, the runtime given above is impractical for applications. We turn our attention to an algorithm which approximately solves the PSD-CC objective, but is much more efficient in practice. We implement a randomized algorithm for sampling vertices of a zonotope (that will eventually enumerate them all), developed by Stinson, Gleich, and Constantine [19]. When mapping a hypercube in  $\mathbb{R}^M$  onto a zonotope in  $\mathbb{R}^N$ , the basic outline of their procedure is as follows. Form an  $N \times M$  matrix  $\mathbf{G}$ , where each column is a generator of the zonotope, i.e.,  $\mathbf{G}$  is the linear map that maps the hypercube into a lower-dimensional space. Given a vector  $\mathbf{x}$  drawn from a standard Gaussian distribution, compute  $\mathbf{v} = \mathbf{G} \text{sign}(\mathbf{G}^T \mathbf{x})$ , where  $\text{sign}(\mathbf{u})$  returns a vector with  $\pm 1$  entries, reflecting the signs of the entries of  $\mathbf{u}$ . The main insight of Stinson, Gleich and Constantine is that under reasonable assumptions on  $\mathbf{G}$ ,  $\mathbf{v}$  will be a vertex of the zonotope. One can construct the entire zonotope by generating vertices in this way, by checking whether a given vertex has been previously found, and continuing until all the vertices have been returned. In practice, it is better to simply approximate the zonotope by stopping after a specified number of vertices have been found.

We alter this procedure slightly to fit our needs. Note that the generators of the signing zonotope come from the outer products of the form  $\mathbf{v}_i \cdot (\mathbf{e}_r - \mathbf{e}_s)^T$ ,

for  $i = 1, 2, \dots, n$  and  $1 \leq r < s \leq d + 1$ . This product gives a  $d \times (d + 1)$  matrix with a zero row sum, so taking the first  $d$  columns and stacking them into a vector we get one of the columns of  $\mathbf{G}$ . Equation (7) shows that when we form a linear combination of these generators, where the coefficients of the linear combination are entries of a signing  $\sigma$ , the output is exactly the  $\mathbf{Z}$ -vector corresponding to  $\sigma$ . We are ultimately interested in extremal signings rather than actual zonotope vertices, so we repeatedly generate vectors  $\sigma = \text{sign}(\mathbf{G}^T \mathbf{x})$ . We then inspect the entries of  $\sigma$  and find the corresponding clustering of  $n$  vectors and thence the clustering's PSD-CC objective score (4). We do this for a very large number of randomly generated extremal signings and output the one with the highest score. We name our algorithm based on this zonotope vertex enumeration, ZONOC, outlined in Algorithm 1.

---

**Algorithm 1** ZONOC

---

**Input:** rows of  $\mathbf{V}$ :  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^d$   
**Step 1** Form generator matrix  $\mathbf{G}$   
**Step 2** {Test  $k$  extremal clusterings:}  
Set  $\text{BestClustering} \leftarrow \emptyset, \text{BestObjective} \leftarrow 0$   
**for**  $i = 1, 2, \dots, k$  **do**  
    1. Generate standard Gaussian  $\mathbf{x} \in \mathbb{R}^M$   
    2.  $\sigma \leftarrow \text{sign}(\mathbf{G}^T \mathbf{x})$   
    3. Determine clustering  $C$  and objective  $C_{\text{obj}}$  from  $\sigma$   
    **if**  $C_{\text{obj}} > \text{BestObjective}$  **then**  
         $\text{BestObjective} \leftarrow C_{\text{obj}}, \text{BestClustering} \leftarrow C$   
**Output:**  $\text{BestClustering}, \text{BestObjective}$

---

Although ZONOC is not guaranteed to return the optimal clustering, Stinson, Gleich, and Constantine prove that with high probability the zonotope vertices that are generated will tend to be those which most affect the overall shape of the zonotope [19]. We expect such extremal vertices of the zonotope to be associated with extremal clusterings of the  $n$  data points, i.e., clusterings with high objective score.

## 5 Numerical Experiments

In this section, we demonstrate the performance of ZONO-CC in a variety of clustering applications. We first illustrate the performance of our algorithm on synthetic datasets that are low-dimensional by construction in order to understand how its behavior depends on the rank, as well as the size of the problem. Second, we study correlation clustering in two real-world scenarios: (i) the volume of search queries over time for computer science conferences and (ii) stock market closing prices for S&P 500 companies. Neither of these cases is intrinsically low-rank. So we study the performance of ZONO-CC on low-rank approximations of the data: curiously the best results are achieved on

extremely low-rank approximations. Our goal in both the synthetic and real-world experiments is to compare our algorithm to other well-known correlation clustering algorithms and, when possible, see how well our algorithm is able to approximate the optimal solution. On the real world-data, we also run the  $k$ -MEANS procedure and find that it is unable to create the clustering we find via correlation clustering.

For our last experiment, we show how to cluster any unsigned network with ZONOC, by first obtaining an embedding of the network’s vertices into a low-dimensional space. We use this technique to cluster and study the structure of several networks from the Facebook 100 dataset [20]. Here we compare ZONOC against  $k$ -MEANS, an algorithm that is very commonly applied to cluster data in low-dimensional vector spaces. The goal of this final experiment is not to show that ZONOC is better in itself, but to analyze the results of ZONOC when both high-quality and low-quality embeddings of the vertices are applied. We find that with lower-quality embeddings, ZONOC generates better quality results than  $k$ -MEANS.

Our experiments revolve around the following four algorithms, three of which are specifically intended for correlation clustering. In our experiments we show runtimes for guidance only and note that these are non-optimized implementations.

**Exact ILP:** For small problems, we compute the optimal solution to the correlation clustering problem by solving an integer linear program with the commercial software Gurobi.

**CGW:** The 0.7664-approximation for maximizing agreements in weighted graphs, based on a semidefinite programming relaxation, by Charikar, Guruswami, and Wirth [5].

**PIVOT:** This is a very fast algorithm developed by Ailon, Charikar, and Newman for  $\pm 1$  correlation clustering instances [1]. It uniformly randomly selects a vertex, clusters it with all nodes similar to it, and repeats. In expectation, it is a 3-approximation. Because of its speed, we return the best running 1000 or 2000 instantiations, depending on problem size.

**$k$ -MEANS:** The standard Lloyd’s  $k$ -means algorithm, as implemented in MATLAB, with  $k$ -means++ initialization [2]. Because of its speed, we return the best of 100 instantiations.

We make code for all of our algorithms and experiments available at <https://github.com/nveldt/PSDCC>.

## 5.1 Synthetic Datasets

We begin by demonstrating that ZONOC computes a very good approximation to the optimal PSD-CC objective even though it does not test all vertices of the zonotope. We use several synthetic datasets for a range of  $d$  values. We consider both the scenario where there is a true planted clustering in the dataset, as well as the case where there is no clear clustering structure in the data. The first of these cases tests each algorithm’s ability to detect a clustering when there is a high signal to noise ratio. In the second experiment we are purely testing how

well each algorithm can optimize the objective in the absence of any particular clustering structure in the dataset. We also compare the performance of PIVOT and ZONOC on larger datasets in a third experiment. Finally, in experiment four we demonstrate how ZONOC performs for a varying number of iterations.

For the first experiment, we perform the following steps to generate a dataset with a planted partitioning:

1. For each rank  $d$ , we set  $n = 10d$  and choose an integer  $k$  between  $d/2$  and  $d + 1$  uniformly at random to be the number of clusters to form.
2. Assign each of the  $n$  objects to one of the  $k$  planted clusters uniformly at random.
3. Assign to each of the planted clusters a vertex of a regular  $d$ -simplex, whose  $d + 1$  vertices in  $\mathbb{R}^d$  all have pairwise negative dot products.
4. Form  $\mathbf{W} \in \mathbb{R}^{n \times d}$  by setting the  $i$ th row to be the coordinates of the simplex vertex assigned to the cluster that object  $i$  belongs to. Now  $\mathbf{W}$  defines a correlation clustering problem with a perfect partitioning.
5. Add noise to form matrix  $\mathbf{V}$  in the following way:

$$\mathbf{V} = (1 - \varepsilon)\mathbf{W} + \varepsilon\mathbf{E}$$

where  $\mathbf{E}$  is an  $n \times d$  matrix of standard Gaussian noise. For our experiments we choose  $\varepsilon = .15$ , which is enough noise to guarantee there will no longer be a perfect clustering, but so that the structure of the planted clustering is still detectable.

After performing the above steps, computing  $\mathbf{A} = \mathbf{V}\mathbf{V}^T$  gives a synthetic dataset with an underlying clustering which will correspond to the optimal clustering or at least approximate it very well. We then run each correlation clustering algorithm on this dataset and compare each algorithm’s objective score against the score of the planted clustering. This setup allows us to test how well each method is able to perform in the high signal to noise regime. In Figure 2 we display objective scores and runtimes for different algorithms on synthetic datasets with planted clusters for a range of  $d$  values from 2 to 20. We use 50,000 iterations of ZONOC and 1000 instantiations of PIVOT in each case.

We note that for this first experiment ZONOC outperforms all other methods for matrices up to rank 15. At this point CGW begins to take over in objective score, although it does so at the expense of a much longer runtime. Recall that we are running ZONOC for a fixed number of iterations; we would expect to see improved objective scores for running the algorithm longer as problem size increases. Note that for even for rank 20, ZONOC still achieves an objective score that is within 85% of the score of the planted clustering.

For the second experiment we generate synthetic datasets without any underlying structure by forming random  $n \times d$  matrices  $\mathbf{V}$  with entries taken from a standard Gaussian distribution. We fix  $n = 60$ , which is small enough so that we

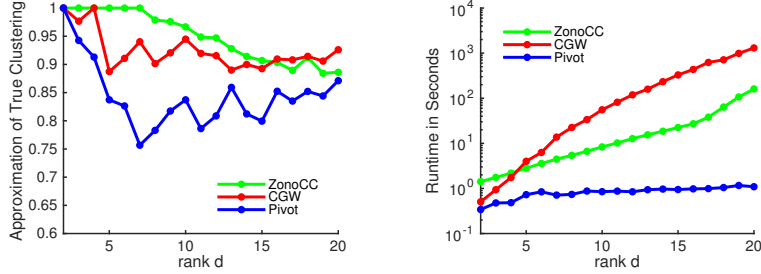


Figure 2: Results for ZONOCC (green), CGW (red), and PIVOT (blue) on synthetic datasets with a true underlying clustering structure. The left plot gives each algorithm’s approximation to the score of the planted clustering for  $d$  ranging from 2 to 20 and  $n = 10d$ . The right plot shows runtimes for each method. For both plots we take the median over five trials. ZONOCC outperforms other algorithms for all values of  $d$  up to 15. At this point CGW finds a better clustering score, but does so at the expense of a much longer runtime. Even for high values of rank, we note that ZONOCC achieves a score that is 85% of the planted clustering.

can obtain the exact solution to the correlation clustering problem for the matrix  $\mathbf{A} = \mathbf{V}\mathbf{V}^T$  by solving the ILP, though at a large computational expense. In this case we are purely testing each algorithm’s ability to optimize the correlation clustering objective function, as there is no true clustering structure to detect. Figure 3 gives a visualization of the approximation ratio for ZONOCC (with 50,000 iterations), CGW, and 1000 instantiations of PIVOT. We give representative runtimes for each method in Table 1. Although we expect this to be a worst-case scenario for our algorithm, we notice that ZONOCC still outperforms the other methods for very low-values of rank. For these experiments PIVOT does extremely well, which we expect is because there are essentially many clusterings that achieve a good score. Just as before, even for relatively high values of rank, ZONOCC is still able to find a clustering that is at least 85% of optimal.

In the third experiment, we wish to understand how ZONOCC compares with PIVOT as we scale the problems up in *size*, for a fixed rank  $d = 5$ . This setting renders both the ILP and CGW methods infeasible, and so we show results only for ZONOCC with 1000 iterations, and 2000 instantiations of PIVOT in Figure 4. These figures show us that ZONOCC always outperforms PIVOT in objective, with the difference slowly growing as the problem size increases.

In the final synthetic experiment, we study how the approximation changes as the number of iterations of ZONOCC increases. We fix the size at  $n = 3000$  and  $d = 5$ . The results in Figure 5 show that the algorithm quickly attains a near-optimal solution, but moves closer to optimality slowly, as it continues to explore more vertices of the zonotope.

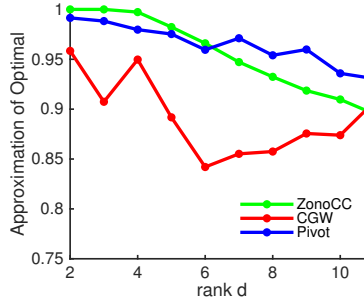


Figure 3: Median approximation ratios for ZONOC (green), PIVOT (blue), and CGW (red) for synthetic datasets generated with no underlying clustering structure. We fix  $n = 60$ , vary the rank  $d$ , and take the median score over five trials. For low values of  $d$ , ZONOC outperforms the other correlation clustering methods, though as  $d$  increases, PIVOT and then CGW take over.

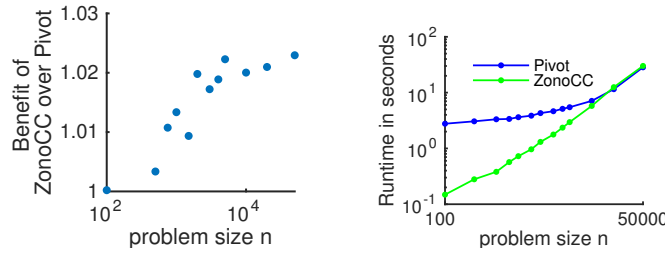


Figure 4: In the left we show the benefit of ZONOC over PIVOT in terms of objective score on synthetic datasets of increasing size  $n$  and rank  $d = 5$ . ZONOC always has a higher objective value, and as problem size grows, so does its performance over PIVOT. On the right we plot corresponding runtimes. ZONOC is faster for problem sizes under  $n = 10000$ . For higher values, times are comparable and scale roughly linearly in  $n$ .

## 5.2 Clustering Using Search Query Data

For our first real-world application we use ZONOC to cluster top-tier computer science conferences based on search query volume data. Each search term is either a conference acronym (e.g., “ICML”), or is an acronym concatenated with “conference” (e.g., “WWW conference”). For each search term, we obtain a time-series of the volume of search queries from Google Trends for each month over the course of a six-year period, 2010–2016.

It is interesting to observe the significance of the clustering returned by ZONOC. In Figure 7 we plot for each cluster the smoothed search query data for all of the conferences in the cluster. We observe that ZONOC effectively partitions the dataset into conferences that have increased in search query volume over the course of the past six years, and those that have experienced an overall



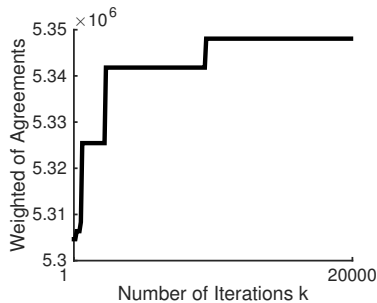


Figure 5: Best objective value as a function of number of iterations of ZONOCCL for a synthetic dataset with  $n = 3000$  and  $d = 5$ . ZONOCCL quickly finds a good-quality clustering, and slowly improves as we let the algorithm run longer.

Table 1: Median running time in seconds for Experiment 2, where we run each correlation clustering method on a dataset with no underlying clustering structure. We set  $n = 60$  and test a range of values for the rank  $d$ .

$d$	Exact ILP	CGW	ZONOCCL	PIVOT
2	2	7	2	2
3	8	8	3	2
4	24	7	3	2
5	50	6	4	2
6	46	6	4	2
8	1060	7	7	2
10	1462	6	7	2

decrease in search volume. We expect it to be unsurprising to our readers that the WWW conference is in the “growing” cluster. We were unable to find any set of three clusters from  $k$ -MEANS that resembles the result of correlation clustering for this problem. The  $k$ -MEANS results tended to generate three clusters of nearly equal size.

We also run ZONOCCL for 50,000 iterations on rank- $d$  approximations of the correlation clustering matrix, where  $d$  ranges from 2 to 15. As  $d$  increases, ZONOCCL tends to form more clusters, but is always able to identify two large groups of conferences that are highly correlated. In terms of the clustering coefficient objective on the original (and not low-rank) matrix, ZONOCCL decreases in performance only because we must maintain a constant number of iterations for the sake of runtime, despite the fact that the number of zonotope vertices increases exponentially in  $d$ . This behavior is illustrated in Figure 6. Even though these are sub-optimal, the clustering returned always has two large clusters and small groups of outliers.

We compare this against running Lloyd’s  $k$ -means clustering algorithm 1000

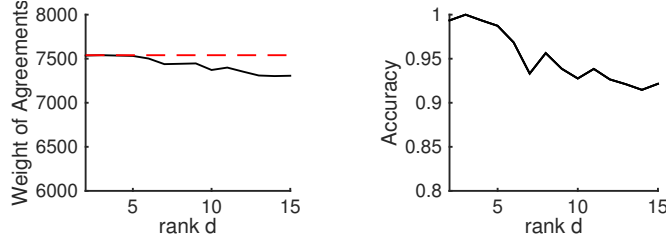


Figure 6: The left plot shows the objective scores achieved by ZONOC for CS conference search query data as we increase rank. The dotted line shows the score of the optimal clustering, which ZONOC remains very close to, regardless of rank. On the right we show the accuracy of the rank  $d$  clustering, which is computed by considering all  $\binom{n}{2}$  pairs in the clustering, and finding the fraction of those pairs for which the decision to cluster together or cluster apart agrees with the optimal clustering of the dataset. In all cases our accuracy is above 90%.

times and taking the best run for the same low-rank approximations. We let  $d$  determine number of clusters for  $k$ -means to form. We evaluate the weight of agreements for each clustering with respect to the original correlation matrix, and find that ZONOC is much more robust than  $k$ -means clustering to changes in the value of  $d$ . The objective scores for both algorithms decrease as  $d$  increases, but the difference is much more marked for  $k$ -means. ZONOC decreases in performance only because we must maintain a constant number of iterations for the sake of runtime, despite the fact that the number of zonotope vertices increases exponentially in  $d$ . As  $d$  increases, ZONOC tends to form more clusters, but is always able to identify two large groups of conferences that are highly correlated. All other conferences are clustered in smaller outlier clusters. On the other hand,  $k$ -means tends to form more evenly blanced clusters for higher values of  $d$ , which is a disadvantage in this case as it tends to separate conferences that are highly correlated in terms of search trends. See Figure 6 for a plot of the sizes of the largest two clusters and the corresponding objective scores for different values of  $d$ .

Given that  $d = 2$  yields the best results for both ZONOC and  $k$ -means, we run both algorithms again to determine how many iterations are needed to reach the best result. We find that we only need three thousand iterations for ZONOC, and a single run of  $k$ -means to consistently return the best clustering. Both procedures take negligible amounts of time. We then compare these results against two other correlation clustering techniques: the Pivot algorithm of Ailon, Charikar, and Newman, and CGW. Pivot is an extremely fast algorithm, so we are able to run it for several thousand trials and take thenclustering with the highest objective score. CGW requires solving an expensive semidefinite program which takes roughly 20 minutes, after which we can apply the randomized rounding step several times and take the best result. In the end, ZONOC, Pivot, and

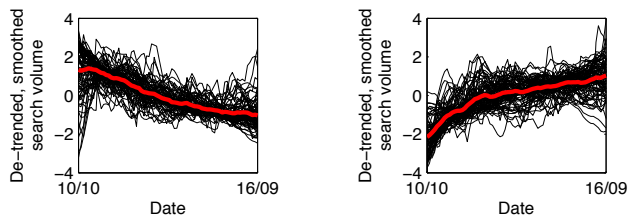


Figure 7: Smoothed search query data for top-tier computer science conferences. The partition discovered by ZonoCC splits the conferences into those whose search volume shows an overall decreasing trend (left), and an increasing trend (right). WWW is in the increasing cluster. For each plot we show an average of the clusters as a thick red line. There is also a third cluster with only one outlier conference that is not shown here.

CGW recover the same clustering with a weight of agreements of 8460.5. Of these algorithms, CGW is by far the slowest. Pivot and ZONOC both consistently recover the best clustering within a second. For  $k$ -means, we only need to run the algorithm once to consistently obtain a clustering with objective score 8457.3. However, no matter how many trials we run or how many clusters we form, we do not improve upon this.

In the end we are able to consistently find the same clustering as ZONOC with Pivot, though we need to run the algorithm for longer. CGW requires running an expensive semidefinite program, and takes over 20 minutes, though after solving the SDP we perform the rounding step of CGW numerous times to obtain the best possible result. We compare objective scores in Table 3. While all values are reasonably close, ZONOC and Pivot are the only two to achieve the highest score, and ZONOC is an order of magnitude faster, although both runtime are small due to the size of the problem.

Table 2: Objective scores and runtimes for four algorithms on search query data. ZONOC is the fastest algorithm to find the best result.

ZONOC	$k$ -means	CGW	Pivot
8460.5	8457.3	8443.7	8460.5
0.1 s	< 0.01 s	> 20 min	1.2s

It is interesting to observe the significance of the clustering returned by ZONOC. In Figure 7 we plot for each cluster the smoothed search query data for all of the conferences in the cluster. We observe that ZONOC effectively partitions the dataset into conferences that have increased in search query volume over the course of the past six years, and those that have experienced an overall decrease in search volume.

Table 3: Objective scores and runtimes in seconds for correlation clustering algorithms on two real-world datasets. We can only use ZONOC and PIVOT on the stocks dataset due to the size of the problem.

Dataset		ZONOC	PIVOT	CGW	ILP
CS Conf. $n = 157$	Obj.	7540.0	7540.0	7540.0	7540.0
	Time	7	1	1380	52
Stocks $n = 497$	Obj.	5100.2	5099.5	—	—
	Time	40	20	—	—

### 5.3 Stock Market Data

The second study we consider is to cluster stock market closing prices on different days of the year. We obtain prices for 497 stocks from the S&P 500 from Yahoo’s finance API over the 253 trading days in a year. We use the correlation between these time series to generate the input to our correlation clustering experiment. In this case we are unable to run the ILP to certify the clustering as optimal. Nor are we able to run the CGW algorithm due to insufficient memory. Thus, we just compare ZONOC against PIVOT: the resulting clusterings are very close, but ZONOC finds the better clustering (see Table 3 for the objectives and runtimes).

Similar to our previous experiment, we discover that there are two large groups of closely correlated stocks and a third cluster with an outlier stock. The outlier is “Public Storage” (PSA), whose stock prices are largely uncorrelated with all other companies. For comparison, running  $k$ -MEANS with 3 clusters always splits up many of the closely correlated companies.

### 5.4 Clustering Social Networks via Embeddings

We can use ZONOC to cluster any dataset where each entry is represented by a vector in a low-dimensional vector space. This means that our algorithm can be used to cluster unsigned network data as long as we have a way to embed the nodes of the network in a low-dimensional space. Such embeddings have been an active area of research recently [8, 17]. We demonstrate how to combine ZONOC with two different graph embedding techniques to produce large clusterings to analyze several networks from the Facebook 100 datasets [20].

The *purpose* of these experiments is not to demonstrate that ZONOC achieves a superior clustering result given the metadata. (Because there is no one algorithm that can achieve this [16].) Rather we wish to compare ZONOC and  $k$ -MEANS – in terms of how their clusters reflect the metadata – on low-quality embeddings from the eigenvectors of the Laplacian and on a high-quality embedding from node2vec [8]

**Datasets.** The datasets we use are subsets of the Facebook graph at certain US universities on a certain day in 2005. These include an undirected graph and

anonymized metadata regarding each person’s student-or-faculty status, gender, major, dorm/residence, and graduation year. We run our experiments on the following networks of different sizes: Reed, Caltech, Swarthmore, Simmons, and Johns Hopkins. We aim to cluster this data based on friendship links in the graph – reflected in the embeddings – and in the process see how the clusterings might be related to different attributes.

**Embeddings.** We consider two different ways to embed each node into a low-dimensional space based on the edge structure of the graph. The first is to take a subset of the eigenvectors of the normalized Laplacian of the network:  $\mathcal{L} = \mathcal{I} - D^{-1/2}AD^{-1/2}$  where  $D$  is the diagonal matrix of node degrees and  $\mathcal{I}$  is the identity matrix. If we take the  $d$  eigenvectors corresponding to the smallest nonzero eigenvalues of  $\mathcal{L}$ , this gives an embedding in  $\mathbb{R}^d$ .

The second embedding we consider comes from an algorithmic framework developed by Grover and Leskovec [8] called *node2vec* for mapping nodes in a network to a low-dimensional feature space for representational learning. The points of the embedding lie in  $\mathbb{R}^d$  for a user-specified  $d$ .

**Results.** For each of the networks studied, we obtain two embeddings into  $\mathbb{R}^3$ , one from the normalized Laplacian and the other using *node2vec*. For each embedding, we center the data by subtracting the mean point. This gives us a set of  $n$  vectors with both positive and negative entries. We then run ZONOC on each embedding, and compare against running  $k$ -MEANS for the same number of clusters as the output from ZONOC.

We analyze our clusterings by observing how the clusters relate to four of the meta-data attributes: student-or-faculty status, major, dorm/residence, and graduation year. The metric we use is the probability that two random people who are clustered together share the same value for the metadata attribute. We can also compute this metric for the entire network to get a baseline score. The results for this experiment are given in Table 4, where the “None” method places all nodes into a single cluster (which is the baseline probability). Note that the only meaningful column across the networks is the *Year* attribute. In addition, Caltech, which is a small school with a strong residential population, shows a similar effect for the dorm attribute. Thus, we focus our attention on the Year attribute.

The table shows that for *node2vec* embeddings,  $k$ -MEANS always gives a higher probability than ZONOC except for Caltech and Johns Hopkins, where they are effectively the same. In contrast, for the embeddings from the Laplacian, the ZONOC always shows stronger alignment with the year attribute. At the very least, this is a demonstration that ZONOC and  $k$ -MEANS can alternate in performance on any given clustering task. However, we suspect that this is evidence that ZONOC is likely to be better in cases with *weak features* (such as the Laplacian).

Table 4: Probabilities that two people in the same cluster share the given attribute. All networks display a strong connection between the clusterings and the graduation year. ZONoCC is better at detecting this trend on the low-quality Laplacian embedding, whereas  $k$ -MEANS performs better on the more sophisticated node2vec embedding.

Network	Emb.	Method	Stud. or Fac.	Major	Dorm	Year
<b>Reed</b>	—	None	0.725	0.037	0.015	0.137
$n = 962$	N2V	ZONoCC	0.698	0.039	0.018	0.278
		$k$ -means	0.756	0.039	0.020	0.325
	Lap	ZONoCC	0.744	0.038	0.018	0.298
		$k$ -means	0.745	0.038	0.018	0.290
<b>Caltech</b>	—	None	0.564	0.063	0.078	0.142
$n = 769$	N2V	ZONoCC	0.576	0.064	0.160	0.151
		$k$ -means	0.566	0.065	0.127	0.145
	Lap	ZONoCC	0.601	0.065	0.087	0.166
		$k$ -means	0.578	0.064	0.080	0.146
<b>Swarthmore</b>	—	None	0.628	0.045	0.049	0.146
$n = 1659$	N2V	ZONoCC	0.620	0.048	0.055	0.262
		$k$ -means	0.627	0.049	0.051	0.265
	Lap	ZONoCC	0.599	0.047	0.042	0.205
		$k$ -means	0.599	0.046	0.042	0.197
<b>Simmons</b>	—	None	0.753	0.043	0.045	0.161
$n = 1518$	N2V	ZONoCC	0.716	0.043	0.064	0.378
		$k$ -means	0.717	0.043	0.065	0.379
	Lap	ZONoCC	0.763	0.044	0.047	0.167
		$k$ -means	0.761	0.044	0.045	0.162
<b>Johns Hop.</b>	—	None	0.618	0.036	0.020	0.134
$n = 5180$	N2V	ZONoCC	0.612	0.041	0.025	0.213
		$k$ -means	0.592	0.041	0.024	0.203
	Lap	ZONoCC	0.632	0.046	0.026	0.229
		$k$ -means	0.608	0.042	0.023	0.187

## 6 Conclusions

Our results introduce a new approach to solving correlation clustering problems by considering the rank and structure of the underlying matrix associated with the problem. This opens a number of new directions in correlation clustering based approaches. The algorithm we present offers a fast and accurate method for solving correlation clustering problems where the input can be represented or at least well-approximated by a low-rank positive semidefinite matrix. We demonstrate a number of applications including clustering time series data from search queries relating to top-tier computer science conferences and stock closing prices. We also demonstrate how this method can be used with embeddings of network data into low-dimensional spaces. We plan to release the code and data

sets used in this investigation for computational replicability once the manuscript is accepted.

In future work we wish to prove more rigorous theoretical approximation results for our methods. For example, can we give an approximation bound for  $k$  iterations of ZONOC. We are surprised to find that PIVOT does well on low rank matrices, and we wish to better understand why. We also wish to understand the impact of low-rank approximation on the clustering coefficient objective.

## 7 Acknowledgments

Nate Veldt is funded by NSF award IIS-1546488, and would like to thank the NSF and the Australian Academy of Sciences for jointly funding his work as a part of the East Asia and Summer Pacific Institute in June-July 2016, during which time much of the above work was performed. Anthony Wirth thanks the Australian Research Council and the Melbourne School of Engineering for funding visits by him to Purdue and David Gleich to Melbourne. David Gleich is partially supported by NSF CAREER award CCF-1149756, NSF award IIS-1546488, NSF STC award CCF-093937, the DARPA SIMPLEX program, and the Sloan Foundation.

## References

- [1] N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):23:1–23:27, 2008.
- [2] D. Arthur and S. Vassilvitskii.  $k$ -means++: The advantages of careful seeding. In *SODA*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
- [3] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.
- [4] A. Bhattacharya and R. K. De. Divisive correlation clustering algorithm (DCCA) for grouping of genes: detecting varying patterns in expression profiles. *Bioinformatics*, 24(11):1359–1366, 2008.
- [5] M. Charikar, V. Guruswami, and A. Wirth. Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383, 2005.
- [6] T. M. Cover and B. Efron. Geometrical probability and random points on a hypersphere. *The Annals of Mathematical Statistics*, pages 213–220, 1967.
- [7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. WH Freeman and Company, New York, 1979.
- [8] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *KDD*, pages 855–864, 2016.
- [9] F. K. Hwang, S. Onn, and U. G. Rothblum. A polynomial time algorithm for shaped partition problems. *SIAM Journal on Optimization*, 10(1):70–81, 1999.
- [10] S. Kim, S. Nowozin, P. Kohli, and C. D. Yoo. Higher-order correlation clustering for image segmentation. In *NIPS*, pages 1530–1538, 2011.
- [11] V. Klee. Separation properties of convex cones. *Proceedings of the American Mathematical Society*, 6(2):313–318, 1955.
- [12] P. P. Markopoulos, G. N. Karystinos, and D. A. Pados. Optimal algorithms for-subspace signal processing. *IEEE Transactions on Signal Processing*, 62(19):5046–5058, 2014.
- [13] A. McCallum and B. Wellner. Conditional models of identity uncertainty with application to noun coreference. In *NIPS*, pages 905–912. MIT Press, 2005.
- [14] M. E. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104:1 – 036104:19, 2006.



- [15] S. Onn and L. J. Schulman. The vector partition problem for convex objective functions. *Mathematics of Operations Research*, 26(3):583–590, 2001.
- [16] L. Peel, D. B. Larremore, and A. Clauset. The ground truth about metadata and community detection in networks. *arXiv*, cs.SI:1608.05878, 2016.
- [17] B. Perozzi, R. Al-Rfou, and S. Skiena. DeepWalk: Online learning of social representations. In *KDD*, pages 701–710, 2014.
- [18] R. A. Rankin. On the closest packing of spheres in  $n$  dimensions. *Annals of Mathematics*, 48(4):1062–1081, 1947.
- [19] K. Stinson, D. F. Gleich, and P. G. Constantine. A randomized algorithm for enumerating zonotope vertices. *arXiv preprint arXiv:1602.06620*, 2016.
- [20] A. L. Traud, P. J. Mucha, and M. A. Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, 391(16):4165–4180, 2012.
- [21] J. Van Gael and X. Zhu. Correlation clustering for crosslingual link detection. In *IJCAI*, pages 1744–1749, 2007.
- [22] X. Zhang and M. Newman. Multiway spectral community detection in networks. *Physical Review E*, 92(5):052808, 2015.